Semi-Supervised Clustering via Learnt Codeword Distances

Dhruv Batra[1] Rahul Sukthankar[2,1] Tsuhan Chen[1]
www.ece.cmu.edu/~dbatra rahuls@cs.cmu.edu tsuhan@cmu.edu

[1]Carnegie Mellon University [2]Intel Research Pittsburgh

Abstract

This paper focuses on semi-supervised clustering, where the goal is to cluster a set of data-points given a set of similar/dissimilar examples. These examples provide instance-level equivalence/in-equivalence constraints (*e.g.*, similar pairs belong to the same cluster while dissimilar pairs belong to different clusters), but in order to aid final clustering we must propagate them to feature-space level constraints (*i.e.*, how similar are two regions in the feature space?). An increasingly popular approach to accomplish this is by learning distance metrics over the feature space that are guided by the instance-level constraints. Inspired by the success of recent bag-of-words models, we utilize codewords (or visual-words) as building blocks. Our proposed technique learns non-parametric distance metrics over codewords from these equivalence (and optionally, in-equivalence) constraints, which we are then able to propagate back to compute a dissimilarity measure between any two points in the feature space. There are two significant advances over previous work. First, unlike past efforts on global distance metric learning which try to transform the entire feature space so that similar pairs are close, we transform modes in data distribution or pockets of the feature space. This transformation is non-parametric and thus allows arbitrary non-linear deformations of the feature space. Second, while most Mahalanobis metrics are learnt using Semi-Definite Programming (SDP), our proposed solution is developed as a Linear Program (LP) and in practice, is extremely fast. Finally, we provide quantitative analysis on image datasets (MSRC, Corel) where ground-truth segmentation is available, and show that our learnt metrics can significantly improve clustering accuracy.

1 Introduction

Traditionally, unsupervised clustering algorithms have been used to 'discover' the structure in the data [9]. However, recent works [10, 11, 20] are beginning to look at semi-supervised clustering where the focus is to allow a user to 'direct' the clustering algorithm to a desired output, through minimal supervision. Interestingly, computer vision has witnessed a parallel in the field of image segmentation, with unsupervised segmentation techniques now primarily used as an initial preprocessing step to generate superpixels [2,7] or

BMVC 2008 doi:10.5244/C.22.90

multiple segmentations [8]. Methods employing varying degrees of supervision have been proposed: from semi-supervised or interactive segmentation through scribbles, strokes or bounding boxes [15, 18] to completely supervised segmentation requiring pixel-annotated ground-truth images [2, 7, 17]. A recent work [12] fits somewhere between the two extremes, and works with partial equivalence constraints.

This paper focuses on semi-supervised clustering, where the goal is to cluster a set of data-points given a set of similar/dissimilar examples. These examples provide instance-level equivalence/in-equivalence constraints (*e.g.*, similar pairs belong to the same cluster while dissimilar pairs belong to different clusters), but in order to aid final clustering we must propagate them to feature-space level constraints (*i.e.*, how similar are two regions in the feature space?). An increasingly popular approach to accomplish this is by learning distance metrics over the feature space which are guided by the instance-level constraints. Inspired by the success of recent bag-of-words models, we utilize codewords (or visual-words) as building blocks. Our proposed technique learns non-parametric distance metrics over codewords from these equivalence (and optionally, in-equivalence) constraints, which we are then able to propagate back to to compute a dissimilarity measure between any two points in the feature space. While this dissimilarity measure is not a valid metric and thus cannot be used with a distance-based clustering technique (*e.g.* kmeans), our experiments confirm that it is a useful measure for an affinity-based clustering technique (*e.g.* normalized cuts [16]).

Semi-supervised distance metric learning has been the focus of a significant number of recent works (*e.g.*, [1, 3, 20, 22]). For a comprehensive overview of distance metric learning and a thorough comparative analysis of recent works, the reader is referred to a few excellent surveys [6, 21]. Xing *et al*. [20] learn a global Mahalanobis distance metric for the feature space, and formulate this problem as a constrained convex optimization, which minimizes the distance between similar data-points, while keeping dissimilar pairs sufficiently far apart. Bar-Hillel *et al*. [1] also learn a Mahalanobis metric with the same objective function but different constraints. While that doesn't change the class of the optimization problem, they show empirical improvements in speed. Goldberger *et al*. (NCA) [5] and Weinberger *et al*. (LMNN) [19] learn Mahalanobis metrics to maximize the classification accuracies achieved by the k-nearest neighbour (kNN) classifier. One significant drawback of all these works is that they learn a global distance metric and the same transformation is applied to the data irrespective of its location in the feature space. In the case of multi-modal distribution of classes, learning such a distance metric can actually result in accuracies worse than those achieved by euclidean distance [22]. More recently, Yang *et al*. [22] and Chang *et al*. [3] have proposed methods that learn local distance metrics.

This paper makes two significant contributions. First, unlike past works on global distance metric learning which try to transform the entire feature space so that similar pairs are close, we transform modes in data distribution or pockets of the feature space. This transformation is non-parametric and thus allows arbitrary non-linear deformations of the feature space. Second, while most of the previous works are formulated as general constrained convex programs or semi-definite programs (SDP), our proposed solution is developed as a linear program (LP) and in practice, is extremely fast. Finally, we provide quantitative analysis on image datasets (MSRC [17], Corel [7]) where ground-truth segmentation is available, and show that our learnt metrics can significantly improve clustering accuracy.

The rest of this paper is organized as follows: Section 2 formalizes the problem statement and presents our approach; Section 3 presents results on synthetic data and segmentation results; and Section 4 concludes the paper with discussions.

2 Proposed Approach

Our problem statement is as follows: given

- a dataset of N points $\{x_i \mid x_i \in \mathscr{R}^p, i \in [N]\}$ (where we define $[N] \stackrel{\text{def}}{=} \{1, 2, \ldots, N\}$),
- a set of similar pairs $\mathtt{S} = \big\{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are similar}\big\}$,
- and (optionally), a set of dissimilar points $\mathtt{D} = \big\{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are dissimilar}\big\}$,

we want to learn a distance metric that minimizes the distance between similar pairs, while keeping the dissimilar pairs sufficiently apart. The next few sections develop the form of distances we work with.

2.1 Codeword Posteriors

The first step in our algorithm to "over-segment" the data, through an unsupervised clustering algorithm. Following the notation popularized by bag-of-words models [13], we refer to these initial set of cluster centres as "codewords". We use a publicly-available implementation of k-means/x-means by Pelleg and Moore [14] to perform this initial unsupervised clustering. Our next step is to compute the posterior codeword distributions, which can be thought of as a soft assignment of a data-point to the cluster centres. Formally, for some data-point i,

$$\pi_i(\upsilon) = \Pr(\upsilon \mid x_i) \tag{1}$$

$$\propto \Pr(x_i \mid \upsilon)\ \Pr(\upsilon) \quad \forall \upsilon \in [k], \tag{2}$$

where the first term is the likelihood of a feature vector given a codeword (υ), and is modelled using an exponential kernel of euclidean distance:

$$\Pr(x_i \mid \upsilon) \propto e^{-\frac{d^2(x_i, \upsilon)}{\sigma^2}}. \tag{3}$$

It should be noted that in the above relation we have overloaded the term υ to be both the index of the codeword, and the corresponding feature vector in $\mathscr{R}^p$. The second term in Equation 2, *i.e.*, the marginal over the codewords could be assumed to be uniform. However, since this quantization is the result of a clustering process, we model this marginal by the observed "popularity" of codewords at the end of the clustering process:

$$\Pr(\upsilon) = \frac{\#\text{members in cluster } \upsilon}{\#\text{data points}}. \tag{4}$$

We can also extract joint codeword posteriors for an ordered pair of data-points (i, j) through an independence assumption:

$$\pi_{ij}(\upsilon_i, \upsilon_j) = \Pr(\upsilon_i, \upsilon_j \mid x_i, x_j) \tag{5a}$$

$$= \Pr(\upsilon_i \mid \upsilon_j, x_i, x_j)\ \Pr(\upsilon_j \mid x_i, x_j) \tag{5b}$$

$$= \Pr(\upsilon_i \mid x_i)\ \Pr(\upsilon_j \mid x_j). \tag{5c}$$

Intuitively, this means that we assume that the soft assignment of one data-point to cluster centres tells us nothing about the way another data-point is assigned.

2.2 Distance metric

We define:

$$\widetilde{\boldsymbol{\pi}}_i \stackrel{\text{def}}{=} [\,\pi_i(1)\ \ldots\ \pi_i(k)\,]^T \text{ and,} \tag{6}$$

$$W = \left(w_{a,b}\right)_{k\times k} \qquad W = W^T, \tag{7}$$

to be the vector holding marginal codeword posterior for data-point i and the matrix holding all possible pairwise distances between codewords. It should be noted, that w_{ab} is not the euclidean distance between these codewords, but a parameter that we would like to learn from data. We describe how these parameters are learnt in the next section, but with the help of such a distance matrix and the codeword posteriors, we can define the "expected distance" between any two points in the feature space as:

$$\mathrm{E}_{\pi_{ij}}\left[\,d(x_i,x_j)\,\right] = \widetilde{\boldsymbol{\pi}}_i^T\, W\, \widetilde{\boldsymbol{\pi}}_j. \tag{8}$$

Consider a random experiment where the data-points (say i,j) are assigned (independently) to codewords (say a,b) with probabilities described by their codeword posteriors (*i.e.* $\widetilde{\boldsymbol{\pi}}_i$ and $\widetilde{\boldsymbol{\pi}}_j$). For any particular trial, the distance between these data-points can be found by looking up the relevant entry in W (*i.e.* w_{ab}). The above expression (Eqn 8) holds the expected distance between these two samples over all such quadratic assignments. It should be noted that while W is a distance matrix, the expression above is not a valid distance metric (because the distance of a point to itself is not necessarily zero), and should just be treated as a measure of dissimilarity. The reason for working with this formulation is that it helps set up an efficient optimization problem to learn these codeword distances. Our experiments (Section 3) confirm that this dissimilarity does in fact capture statistics consistent with the provided equivalence/inequivalence constraints.

2.3 Setting up the Linear Program

We would like to learn this distance matrix W such that the distance between similar pairs (*i.e.*, the pairs in $\mathtt{S}$) is minimized. Clearly, some constraints are required to keep the entire dataset from collapsing onto a point. In a manner similar to the formulation of Xing *et al.* [20], we constrain this problem by requiring the distance between all dissimilar pairs to be greater than 1 (any non-trivial distance metric that violates this property can just be scaled to become feasible, while still remaining a valid metric). In order for W to actually be a distance matrix, we also need other constraints: symmetry, non-negativity and triangle inequality.[1] Overall, our optimization problem may be written as:

$$\min_{W} \sum_{(x_i,x_j)\in\mathtt{S}} \widetilde{\boldsymbol{\pi}}_i^T\, W\, \widetilde{\boldsymbol{\pi}}_j \tag{9a}$$

$$\text{s.t.} \quad \widetilde{\boldsymbol{\pi}}_i^T\, W\, \widetilde{\boldsymbol{\pi}}_j \geq 1, \forall\,(x_i,x_j)\in\mathtt{D} \tag{9b}$$

$$W = W^T,\, w_{ii} = 0,\, w_{ij} \geq 0 \tag{9c}$$

$$w_{ij} + w_{jk} \geq w_{ki}, \forall\,(i,j,k). \tag{9d}$$

[1] Strictly speaking, we learn pseudo-metrics not metrics, since w_{ij} may achieve its lower bound (of 0).

The objective function and all constraints are linear in W and thus this problem can be solved by standard LP solvers. If a set of dissimilar pairs (D) is not explicitly known (*e.g.*, in a partial equivalence knowledge scenario [12]), then all pairs belonging to different codewords are used for the constraints in Equation 9b. Overall, this program has $O(k^2)$ variables (or $\binom{k}{2}$ free variables), $O(N^2)$ lower bounds on distance between data-points in the worst case, and 3 $\binom{k}{3}$, or $O(k^3)$ triangle inequality constraints. Thus the size of this optimization problem scales quadratically with the size of dataset, and cubically with the number of codewords. In practice, very few codewords (5 – 30) are required, and off the shelf LP solvers (like CPLEX) are fast enough.

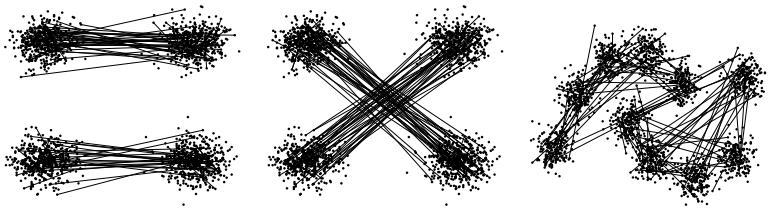
3 Experimental Setup and Results

3.1 Synthetic Data

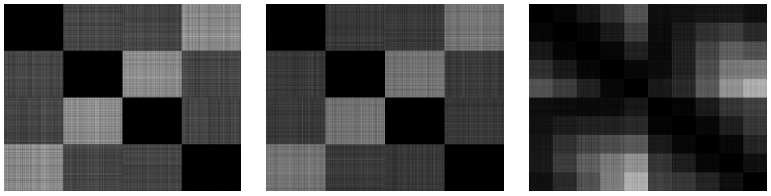
In order to better understand this algorithm, we first report results on synthetic data: "Four Gaussians", and "Two Moons", as shown in Figure 1. The number of codewords (chosen automatically by the x-means algorithm [14]) for these datasets was 4 and 10, respectively. For the "Four Gaussians" dataset, two experiments were conducted: in the first, the two clusters in the same row were considered 'similar'; in the second, the two diagonal clusters were considered 'similar' (XOR problem). For the "Two Moons" dataset, the goal was to discover the two moons as two different clusters. The set of similar points for all three experiments was constructed by randomly choosing 1% of all possible pairs, and our distance metric was learnt as described above. The normalized cuts algorithm [16] was used to generate two clusters using our learnt dissimilarity measure, and the euclidean distance. Figure 1 shows the final clustering results achieved using both distance metrics for these three experiments. For the first two experiments (on the "Four Gaussians" dataset), euclidean distance clustering results oscillate between the one shown in row 4 of Figure 1 and its symmetric form. Thus, euclidean distance can be expected to work half the time for the first task (same row), but never for the second task (XOR problem). In the "Two Moons" dataset, euclidean distance results in confusion in the centre region. Our learnt distances result in the correct clustering for all three experiments. It should be noted that no global linear transformation of the feature space (*e.g.*, those used by [1, 5, 19, 20]) can be successful in the second and third experiments.

3.2 Semi-Supervised Segmentation

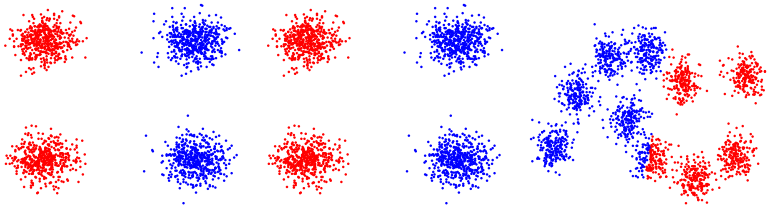
We perform quantitative evaluation of our method for the task of semi-supervised segmentation. We work with the 21-class MSRC [17] and the 7-class Corel [7] datasets, and assume that the "true clustering/segmentation" is given by the class labels. Pixel-level ground-truth annotations for these datasets are available and this allows us to measure the performance of our method in terms of clustering accuracy. Our experimental setup is as follows: given an input image, we extract superpixels using normalized cuts [4]. Thus, an image represented as a collection of superpixels, forms the equivalent of a dataset of points $\{x_i \mid i \in [N]\}$ from our discussion in section 2. The reason for working with a collection of superpixels instead of pixels is that it ensures a locally smooth assignment, and also speeds up the algorithm. The features extracted from these superpixels are simple colour features (average RGB and HSV). We randomly split this collection of superpixels



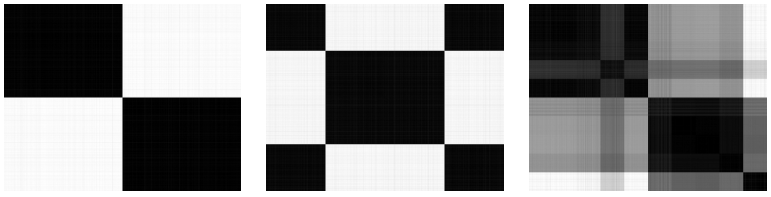
(a) Data + constraints (b) Data + constraints (c) Data + constraints



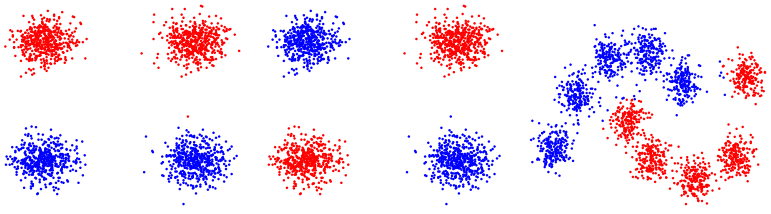
(d) L_2 distances (e) L_2 distances (f) L_2 distances



(g) L_2 clustering (h) L_2 clustering (i) L_2 clustering



(j) Learnt expected distances (k) Learnt expected distances (l) Learnt expected distances



(m) Learnt clustering (n) Learnt clustering (o) Learnt clustering

Figure 1: [Best viewed in colour] Synthetic data results: The first two columns correspond to the "Four Gaussian" dataset (with different similarity constraints); the third column corresponds to the "Two Moons" dataset. Only a subset of the similarity constraints are shown.

into training, validation and testing sets, while maintaining similar distribution of classes. All possible pairs of superpixels belonging to the same class from the training set are provided as input (S) to our algorithm and distance metrics are learnt. In order to compare our proposed method with current state-of-art techniques, the following experiments were performed:

Euclidean (Euc) + Ncut. In this experiment, the set S was ignored, and distance between two data-points was simply defined to be the euclidean distance between their corresponding feature vectors. The normalized cuts algorithm (based on this distance matrix) was then used to cluster the data-points (superpixels).

LMNN + Ncut. For this experiment, we learn distances using the Large Margin Nearest Neighbour (LMNN) algorithm proposed by Weinberger *et al*. (LMNN) [19]. It should be noted that this algorithm requires more information/supervision than the other methods. It needs labels for all training data-points, and thus cannot be used for problems with partial equivalence constraints (*e.g*., [12]). From the point of view of a scribble-based interactive segmentation system, this means that in addition the scribbles, a user would also be required to provide a mapping from the scribbles to cluster/object IDs. For this experiment, the training set of superpixels (along with their ground-truth labels) were provided to the algorithm, and a Mahalanobis distance over this feature space was learnt. Normalized cuts (based on this distance matrix) was used to cluster all the data-points.

Xing + Ncut. In this experiment, the set S was provided as input to the method proposed by Xing *et al*. [20] and a Mahalanobis metric was learnt. Normalized cuts (based on this distance matrix) was used to cluster the data-points.

NPCD + Ncut. In this experiment, non-parametric codeword distances (NPCD) were learnt using the proposed algorithm, and normalized cuts based on the expected distance matrix was used to cluster the data-points.

For all of the above experiments, the affinity matrix (required by Ncut) was generated via an exponential kernel over the distance/dissimilarity measure. To measure clustering performance, we use the clustering accuracy metric proposed by Xing *et al*. [20]. For a two-cluster clustering problem, it is defined as:

$$\text{Accuracy} = \sum_{i>j} \frac{\mathbf{1}\{\mathbf{1}\{c_i == c_j\} == \mathbf{1}\{\hat{c}_i == \hat{c}_j\}\}}{0.5\,n\,(n-1)}, \tag{10}$$

where $\mathbf{1}\{\cdot\}$ is an indicator function that is 1 if its input argument is true (and 0 otherwise), c_i and $\hat{c}_i$ are the true and predicted cluster labels for data-point i, and n is the number of testing samples. Basically, this metric gives an estimate of the probability that for two randomly drawn points, our clustering agrees with the ground-truth clustering (on whether these points are in the same or different clusters). As discussed by Xing *et al*. [20], in the case of multi-class clustering, this metric tends to give inflated scores because a priori most pairs will be in different clusters, and almost any clustering will predict that. Thus, we report normalized clustering accuracies which weigh inter-cluster and intra-cluster clustering accuracies equally. As described above, the similarity and dissimilarity relationships are taken from a subset of the superpixels in each image (training set), a disjoint subset is used as a validation set and the accuracy measure (Eqn. 10) is computed only over pairs from the remaining superpixels (test set). Each image is treated independently; the training set from an image only affects the processing of that single image. The two parameters in our algorithm: number of codewords k, and scaling coefficient in the exponential kernel σ are chosen such that validation set accuracy is maximized. For both

	Clustering Accuracy (%)			
	Euc + Ncut	Xing + Ncut	LMNN + Ncut	NPCD + LMNN
MSRC	56.5 $\pm$ 0.7	69.7 $\pm$ 0.7	67.8 $\pm$ 0.7	**72.9** $\pm$ 0.7
Corel	59.5 $\pm$ 1.1	74.0 $\pm$ 1.1	69.8 $\pm$ 1.1	**75.7** $\pm$ 1.1

Table 1: Clustering accuracies for the 21-class MSRC and the 7-class Corel datasets. NPCD significantly outperforms baseline methods.

(a) 2% (b) 6%

Figure 2: Effect of annotations on size of S

the datasets, we work with a random train-val-test split of (20%-40%-40%), which results in the set S containing approximately 2% of all pairwise constraints (that could be constructed among the given set of superpixels). Figure 2 help us develop an intuition for the scale of this ratio, by showing two scribbled images, and the size of the resultant similarity set S. We note that this level of annotation can be provided by a user with minimal effort through an interactive system.

Comparison. Table 1 compares the performance of our proposed non-parametric codeword distances (NPCD) with these baselines. For both the datasets (MSRC and Corel), test set clustering accuracy averaged over all images is reported, and 95% confidence intervals were determined through analysis of variance (ANOVA). We can see that on both the datasets, our proposed method outperforms all three baselines. Figure 3 shows a few example segmentations achieved by our method, compared to the euclidean distance based segmentations. We can see that in the case of multi-coloured objects (which can be thought of as a multi-modal distribution in colour space), euclidean distance based segmentation tends to cut along colour boundaries which are in fact not object boundaries. Our method, on the other hand, is able to learn, *e.g.* in the case of the 'sign' in row 1, that the distance between 'blue' and 'white' is low and is thus able to keep them both in the same cluster, while separating them from the 'brown' regions of the background. A similar example is shown in row 6, where euclidean distance based segmentation is unable to keep 'black' regions of the sheep (head and feet) in the same segment as the rest of its body, because it is understandable that those two colours are far in the feature space. Our algorithm, however, can learn that those two regions of space are actually close by, and thus does a better job of segmenting the sheep.

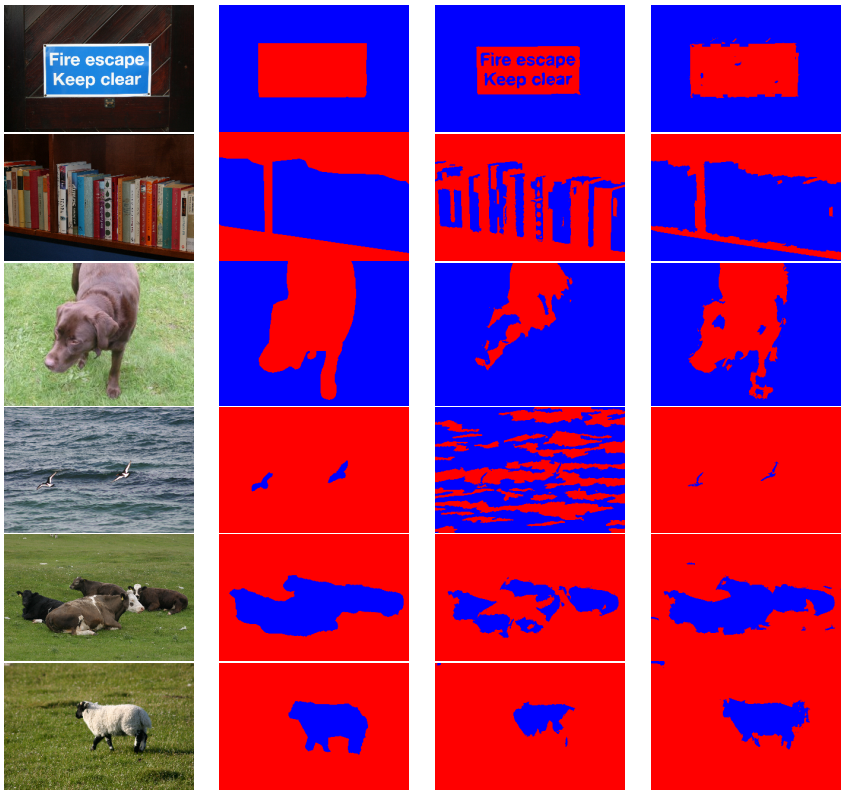


Figure 3: [Best viewed in colour] Example segmentation results on MSRC: column 2 shows the ground-truth segmentaiton; column 3 shows segmentations achieved by euclidean distances; and column 4 shows segmentations achieved by our learnt distances.

4 Conclusions

We focus on semi-supervised clustering, where unlike unsupervised clustering, the goal is not to 'discover' structures in the data, but to develop tools that would allow a user to guide the clustering algorithm towards a desirable output with minimal input. We develop a new algorithm for this task by learning non-parametric distance metrics between codewords (in the feature space). Unlike past works on global distance metric learning, which learn a global linear transformation of the feature space, we transform modes in data distribution or pockets of the feature space. This transformation is non-parametric and thus allows arbitrary non-linear deformations of the feature space. On synthetic data, we were able to visualize the need for such non-linear transformations (XOR task and Two Moons), and show how our method is successfully able to handle these cases. We also pose interactive segmentation as a semi-supervised clustering problem, and show that our method outperforms state-of-art metric leaning techniques on standard datasets (MSRC, Corel).

Acknowledgments

We thank Geoff Gordon, Carlos Guestrin, Devi Parikh, and Yaser Sheikh for helpful discussions, and acknowledge the computing resource support from Intel Research Pittsburgh and the VMR Lab.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *JMLR*, 6:937–965, 2005.

[2] D. Batra, R. Sukthankar, and T. Chen. Learning class-specific affinities for image labelling. In *CVPR*, 2008.

[3] H. Chang and D.-Y. Yeung. Locally smooth metric learning with application to image retrieval. *ICCV*, 2007.

[4] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *CVPR*, 2005.

[5] J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *NIPS*, 2004.

[6] N. Grira, M. Crucianu, and N. Boujemaa. Unsupervised and semi-supervised clustering: a brief survey. *MUSCLE European Network of Excellence*, 2004.

[7] X. He, R. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, 2006.

[8] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, 2005.

[9] D. Jiang, C. Tang, and A. Zhang. Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

[10] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML*, 2002.

[11] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: a kernel approach. In *ICML*, 2005.

[12] S. Kumar and H. Rowley. Classification of weakly-labeled data with partial equivalence relations. *ICCV*, 2007.

[13] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *ECCV*, 2006.

[14] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, 2000.

[15] C. Rother, V. Kolmogorov, and A. Blake. "Grabcut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[16] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22(8):888–905, 2000.

[17] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[18] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum. Background cut. In *ECCV*, 2006.

[19] K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*. 2006.

[20] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *NIPS*. 2003.

[21] L. Yang and R. Jin. Distance metric learning: A comprehensive survey. *Technical Report, Michigan State University*, 2006.

[22] L. Yang, R. Jin, R. Sukthankar, and Y. Liu. An efficient algorithm for local distance metric learning. In *AAAI*, 2006.